# Modeling quantitative requirements in SLAs with Network Calculus[*]

Sebastian Vastag
Informatik IV, TU Dortmund
D-44221 Dortmund
Germany
sebastian.vastag@udo.edu

## ABSTRACT

When planning Service-Oriented Architectures requirements declared in Service Level Agreements (SLAs) have to be considered. SLAs cover functional as well as quantitative requirements like load levels, services rates and delay times. As external factors can influence distributed systems, SLAs have to include tolerances for quantitative requirements.
Early design phases of SOA use analytic models to check functional properties. However, formalization of quantitative requirements in SLAs and their validation in analytic models is still a field of research. A challenge is the description of soft deadlines and the way delay times grow under different load levels.
Network Calculus system theory can give bounds on delay times in systems. It has already been used to validate hard deadlines in networks and embedded systems. For its use in SOA models, soft deadlines and other aspects derived from SLAs have to be included. This paper introduces a new method to control delay times in Network Calculus models in order to specify quantitative requirements. The basic Network Calculus concept of arrival and service curves is extended with delay curves and their relationship is discussed.

## Categories and Subject Descriptors

C.4 [**Computer Systems Organization**]: Performance of systems—*Modeling techniques*

## General Terms

Theory, Performance

## 1. INTRODUCTION

Large-scale distributed systems like Service-Oriented Architectures (SOA) are composed of independent software systems interacting over a network. Each component offers a capability such as processing power, data storage or user authentication as a service to the system. In a networked world, these system nodes can be hosted at different service providers with individual bias in quality, speed and service. The performance of compound systems dependeds on these qualities.
As SOA systems take an important role in business applications they have to meet requirements in availability and response times. These requirements are laid down in Service Level Agreements (SLA) [2, 10] being part of contracts between customer and service provider.

Planning and setting up an conceptual design for a SOA with respect to a given SLA involves the use of models. Especially in early design phases analytic modeling methods are used. They should offer the system engineer the ability to validate the abstract system model against the SLA or to readjust the selection of service providers to the requirements.
Two types of properties can be found in SLAs. The first are functional properties like data formats and interface descriptions. The second type of properties are the non-functional or so-called quantitative requirements. They include maximal load and minimal service level, response times and availability of services [2].

Main challenges in SLA validation with analytic models is the formalization of those non-functional properties in a model and their validation in the analysis step. As the performance and availability of SOAs depends on the load, network traffic and other non-influenceable conditions their behaviour seems to be stochastic. For this reason, SLAs allow some flexibility in deadlines and performance numbers. For example, an SLA includes a soft deadline condition. A system is compliant with the SLA if at least 80% of all requests are served in a specified time interval. Requests may take longer due to unexpected conditions and events.

Usage of soft deadlines makes the case of SLA validation in models of service-oriented architectures different to other modeling domains. Queuing theory is widely used for system modeling. It deals with arrival and service rates and the resulting delays. The downside is that performance numbers are average values that are to inexpressive for validating soft deadlines in SLAs. For modeling performance bounds in data networks Network Calculus [3] can be used. The strongly related Real-Time Calculus [7, 8, 9] is utilized in realtime system design. Both calculi use (min,+)-algebras [1] and characterize conditions on request arrival and service

[*]This research was supported by the Deutsche Forschungsgemeinschaft (DFG).

VALUETOOLS 2011, May 16-20, Paris, France
Copyright © 2011 ICST 978-1-936968-09-1
DOI 10.4108/icst.valuetools.2011.245735

rates with a specific set of functions giving upper and lower bounds on these values. These so-called curves are focussed on maximum execution times for requests to comply with hard deadlines. Open research issues in analytic SLA validation are the selection of suitable modeling methods for SOA and the formalization of quantitative requirements. In optimal case the transformation can be done without loss of information and can be checked for performance numbers.

The first contribution of this work is an algebraic method to describe many non-functional requirements in SLAs concerning timing and delays with (min,+) based calculi. In addition to arrival and service functions a new function set is introduced to Network Calculus or Real-Time Calculus. Delay curves allow definition of upper bounds for processing times in systems. They bring the flexibility to Network Calculus for modeling soft deadlines and other quantitative requirements regarding time behavior in SLAs. Because now delays can be described similar to arrival and service rates they integrate well into (min,+) algebraic systems. This is an important aspect to support analytic validation of SLAs with quantitative requirements.
Next to validation, quantitative requirements in SLAs can be used for capacity planning and service provider selection for SOA systems. As a second contribution the new description of delays is used to find lower bounds on service rates a service provider can deliver necessary to comply with an SLA. This is done by setting up an optimization problem to approximate the smallest service curve fulfilling the requirements given by an SLA with delay curves.

Modeling of SOA including SLAs has already been done with simulation models [5]. In [2] process models are simulated to validate SLAs with statistic methods. A downside of simulation is that models have to be very detailed to replicate real system performance. This makes them not the first choice for evaluating early system designs. Response times of systems in general can also be evaluated with analytic approaches. Interval Timed Coloured Petri Nets [11, 12] or pure Network Calculus [3, 4] give worst-case execution times. As stated above, this may not always be sufficient to model SLAs in detail.

A short summary of (min,+)-calculus is given in Section 2 and its application for the specification of arrival and service rates. Section 3 introduces the new concept of delay curves as a natural extension of Network Calculus and Real-Time Calculusmodeling methods. Delay curves are used in section 4 to approximate the minimum service curve that complies with a given specification. Numerical examples are given in Section 5.

## 2. MIN-PLUS CALCULUS

The so-called min-plus or (min,+)-algebra belongs to the family of tropical algebras [3]. It is an algebraic approach to analyze models of discrete event-systems. Extensive fundamental work has been done in [4, 1].
(min,+) is created from standard algebra by using the minimum function as additive operation and replacing the multiplicative operation with an addition. In other words, $(+, \cdot)$ becomes (min,+). As their original counterparts the operator min() and + form a dioid [3] with $\infty$ as neutral element of addition and 0 as neutral element of multiplication. For example, the term $(5 + 0) \cdot (2 + 4)$ is expressed in (min,+) as $\min(5, \infty) + \min(2, 4)$.
Two more specialized methods have been derived based on the (min,+) linear system theory of Baccelli et. al. [1]. Network Calculus allows the modeling and analysis of network performance behaviour [3]. Thiele et. al. used (min,+)-algebras for Real-Time Calculus to describe execution bounds in embedded systems [7, 8, 9]. Both approaches are strongly related and use the same basic theory for different applications. Before the new concept of delay curves for SLA formalization is presented some common concepts of Network Calculus and Real-Time Calculus are introduced.

### 2.1 (min,+) Convolution

Convolution is an operation between two functions. The result is a new function forming the overlay of both functions. It plays an important role in mathematics and natural sciences. In radio transmissions for example, the folding of carrier frequency and load signal can be expressed by convolution. Applying filters to digital photos corresponds to the folding of picture and filter as two-dimensional functions. Algebraic systems based in (min,+)-algebra can also make use of convolution to work with functions. A main operation in Network Calculus and Real-Time Calculus is (min,+)-convolution. It is used to derive the effective performance for systems out of information on arrival and service rates. In classic algebraic systems convolution is an integrative sum over a product of two real-valued functions $f$ and $g$:

$$(f \circ g)(t) = \int_{-\infty}^{+\infty} f(t-s)g(s)\, ds \tag{1}$$

In (min,+) convolution is restricted to wide-sense increasing functions [3]:

*Definition 1.* A function is wide-sense increasing if and only if $f(a) \leq f(b)$ for all $a \leq b$. $\mathcal{G}$ is the set of wide-sense increasing functions with $f(t) \geq 0\, \forall t, f \in \mathcal{G}$. $\mathcal{F}$ is the subset of $\mathcal{G}$ with functions that are zero for $t < 0$.

As mentioned above, in (min,+) additive operators are exchanged for the minimum function. The integral becomes a minimum or, to allow non-continous input functions, the infimum [3].

*Definition 2.* Let $f$ and $g$ be two functions or sequences in $\mathcal{F}$. The (min,+) convolution of $f$ and $g$ (notation $f \otimes g$) is the function

$$(f \otimes g)(t) = \inf_{0 \leq s \leq t} \{f(t-s) + g(s)\}$$

If $t < 0 : (f \otimes g)(t) = 0$.

(min,+)-convolution is closed in $\mathcal{F}$. For additional properties we refer to chapter 3 in [3].

### 2.2 System Model

The basic system model in Network Calculus and Real-Time Calculus are similar to queuing systems. Workload arrives at a system node and awaits service by the system. After processing ended it leaves the system. Workload can be either computation tasks, customers, data packets or anything

else the model specifies. In the following, jobs or request are pooled to the term arrivals.
This introduction uses a fluid system model with continuous time domain. Although arrivals in real systems are discrete objects they are abstracted to continuous flow. The delivered service towards the arrivals is also included into the flow model. The advantage of continuous models is their simplified notation. As there exists a mapping [3] between discrete and fluid models both methods can be exchanged.

Figure 1 contains the basic system model on the left. As usual in abstract queuing notation arrivals ($R$) enter the system from the left and leave after they have been served to the right ($R^*$). Service in the system itself consumes limited resources arriving from the top ($C$). If there are unused resources left they leave the system ($D$).

## 2.3 Arrival and Service Curves

Network Calculus relies on functions describing arrivals over time. The rate of the arrival flow at time $t$ can be expressed by the slope of a function $a(t)$. Then the amount of arrivals in interval $[0, t]$ is:

$$R(t) = \int_0^t a(x)\,dx$$

$R(t)$ is continuous, wide-sense increasing and $R(t) = 0$ for $t <= 0$, thus $R(t) \in \mathcal{F}$. In Network Calculus, $R(t)$ is called arrival function. For the system node in Figure 1, $R(t)$ is the arriving workload.
Arrival functions are expressions of the underlying arrival flow, they can be derived from real world measurements or simulation results. To characterize arrival flows and to set bounds on arrival rates, Network Calculus abstracts individual arrival functions with new functions called arrival curves conforming to the arrival curve property [9]. It includes curves for limiting minimum and maximum arrivals.

*Definition 3.* An upper arrival curve $\alpha^U$ or a lower arrival curve $\alpha^L$ satisfies the relation

$$\alpha^L(t-s) \leq R(t) - R(s) \leq \alpha^U(t-s) \quad \forall\, 0 \leq s \leq t$$

Network Calculus references $\alpha^U$ as arrival curve, $\alpha^L$ as minimum arrival curve while Real-Time Calculus mentions them as request curves. Figure 2 includes arrival function $R(t)$ that is bounded from above by arrival curve $\alpha^U$.

Although every $f \in \mathcal{F}$ can serve as arrival curve only a small group of basic functions is used. Most of them are linear or grow stepwise and can be combined to more complex functions. A catalog of common functions can be found in [3]. To describe arrival flows bounds of data network packets Network Calculus and Real-Time Calculus make use of the T-SPEC traffic specifications [6].

T-SPEC functions describe arrival bounds for variable bit rate connections (VBR) [3] by piecewise linear approximation:

$$\text{T-SPEC}(t) = \min(M + pt, rt + b) \tag{2}$$

The two affine functions are used to describe separate situations for the arrival flow. The first function $M + pt$ limits an fast arrival flow that can be handled for a short time. $M$ is maximum packet size the network can transfer and $p$ is the peak arrival rate. The second function $rt + b$ sets bounds for long term arrivals: $r$ is the sustainable arrival rate and $b$ is the burst tolerance.. Instances of T-SPEC specifications are characterized by tuple $(p, M, r, b)$. As fluid systems have no packet size we will apply $M = 0$ in general.
Thiele et al. [7] suggested to use three segments but for many applications the combination of two affine functions is sufficient. The examples in figures 2, 3 and 4 include $\alpha^U$ as T-SPEC arrival curve.

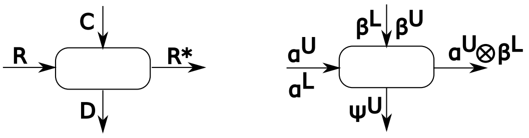

**Figure 1: System Model with functions and curves.**

In the same way arrivals to a system can form arrival functions the amount of delivered work towards arrivals can be expressed with service functions. Work or service performed by the system can be measured in units.
The slope of function $b(t)$ describes the rate of used work units at the point of time $t$. Function $b(t)$ describes the so-called flow of service.
For the amount of service in interval $[0, t]$ we have the service function

$$C(t) = \int_0^t b(x)\,dx$$

The cumulative function $C(t) \in \mathcal{F}$ is the amount of work applied to arrivals by a system in interval $[0, t]$. The system model in Figure 1 handles $C(t)$ as a second input for processing. Abstraction from arbitrary service functions is done with minimum and maximum functions satisfying the service curve property:

*Definition 4.* An upper service curve $\beta^U$ or a lower service curve $\beta^L$ for a service function $C(t)$ is given by the relation:

$$\beta^L(t-s) \leq C(t) - C(s) \leq \beta^U(t-s) \quad \forall\, 0 \leq s \leq t$$

Upper service curve $\beta^U$ sets an upper bound on the service capacity of a system: the sum of performed work units up to time $t$ will never exceed $\beta^U(t)$. Just as well the system will always deliver at least $\beta^L(t)$ work units until $t$.
The system model in figure 1 uses both service curves as input. In figure 2 the relationship between service function $C(t)$ and lower bound $\beta^L$ is visible. For service curves every wide-sense increasing function can be used, a set of functions can be found in [3].
A common function for service curves is the rate-latency function $\beta_{R,T}(t) = \max\{0, R(t-T)\}$. Parameter $R$ is the sustainable service rate that can be continuously delivered, $T$ is the system latency as the maximum time with no service at all. Figures 2, 3 and 4 include $\beta^L = \beta_{R,T}$. In contrast to $\alpha^U$, service curves begin with low service rates. In this work we will focus on the upper arrival curve $\alpha^U$ and the lower service curve $\beta^L$. Theorems and applications for the complete curve set can be found in [3, 9, 8, 7].

## 2.4 Departure curves

When arrival flow $R(t)$ enters a system it is served up to available service $C(t)$. If $R(t) > C(t)$ a part of the arrival flow has to be enqueued. Backlogged arrivals are processed again when $C(t)$ exceeds $R(t)$ again.

Incoming and backlogged leave the system, in terms of Network Calculus this is called the outgoing arrival flow. When we also cumulate the outgoing flow in interval $[0, t]$ we get departure function $R^*(t)$ [3], compare Figure 1. As $R^*(t) \in \mathcal{F}$ has arrival curve properties and can be used to feed a second system node it is also called outgoing arrival curve in Network Calculus [3]. Figure 2 shows the outgoing arrival function $R^*(t)$ as dashed line.

At least two interesting system quantities can be derived. The first one is amount of backlogged arrivals waiting for service at time $t$: $R(t) - R^*(t)$. Buffer sizes for system can be dimensioned adequate by computing the maximum backlog.

The second one is the maximum turnaround time for arrivals entering the system at time $t$. Delay is the additional time $R^*(t)$ needs to draw level with $R(t)$:

$$d(t) = \inf\left\{\tau \geq 0 : R(t) \leq R^*(t+\tau)\right\} \tag{3}$$

Results from Network Calculus analysis do not represent expectation values as found in queuing theory. Depending on the input curves, either best-case or worst-case numbers are computed. The accuracy of these bounds depends on the matching of input curves to the real world system.

An important ability of Network Calculus system theory is to calculate the lower envelope of all output functions $R^*(t)$ just with the input of an arrival function $R$ and an lower service curve. This advantage comes from the usage of (min,+) convolution [3].

$$R^*(t) \geq \inf_{s \leq t}(R(s) + \beta^L(t-s)) = (R \otimes \beta^L)(t) \tag{4}$$

The example in figure 3 is extended with a theoretic lower service curve $\beta^L$ instead of $C(t)$. The dashed line is the resulting departure curve $R^*(t) = R(t) \otimes \beta^L(t)$ when the system delivers the slowest possible service grade. The convolution applies $\beta^L$ at every point of $R(t)$ and obtains the minimum. In figure 3 the convolution is illustrated by moving $\beta^L$ along $R(t)$.

For system modeling and performance evaluation are worst-case scenarios significant. Maximum reachable load has to be combined with the smallest possible service rates. In Network Calculus this situation is expressed with upper arrival curves as arrival functions $\alpha^U = R(t)$ and $\beta^L = C(t)$ for equality between service function and curve. With given upper arrival curve $\alpha^U(t)$ and a lower service curve $\beta^L(t)$ the worst-case departure curve is formed from equation 4 when $R(t) = \alpha^U(t)$:

$$\alpha^{L\prime}(t) = (\alpha^U \otimes \beta^L)(t) \tag{5}$$

For this worst-case scenario a third example is given in figure 4. $\beta^L$ is folded with $\alpha^U$ and forms $\alpha^{L\prime}$ as the lowest of all possible departure curves for this system.

Now we can make statements on the system delay under high load and bad service conditions. The outgoing arrival curve $R^*(t)$ is replaced with upper departure curve $\alpha^{L\prime}(t)$. The virtual delay for a job arriving at time $t$ under worst-case conditions is

$$d(t) = \inf\left\{\tau \geq 0 : \alpha^U(t) \leq \alpha^{L\prime}(t+\tau)\right\} \tag{6}$$

# 3. DELAY CURVES

In section 2.3 arrival and service functions are introduced as cumulated sum of their underlying arrival and service flows. Curves set lower and upper bounds for those functions.

The central new concept presented here is to handle occurring delays as flow. Whenever the arrival function is above the service function, $R(t) > C(t)$, arrivals face a delay because of backlogging. Similar to arrivals and service a flow of measured delays is formed for any point of system time. This delay flow as a function of time can be integrated and form a delay function.

For formalization of quantitative requirements in SLAs, delay functions are bounded with delay curves. The overall delay experienced by arrivals to the system can be described over time. There is no need to set individual maximum delays for arrivals, instead the aggregated delay of all arrivals is limited. This modeling method gains more flexibility than using just hard deadlines.

We start with the definition of the delay function. The virtual delay $d(t)$ for arrivals at time $t$ is given in equation 6. For the total delay in time interval $[0, t]$ we integrate $d(t)$:

*Definition 5.* Let $d(t)$ be the delay between an arrival curve and an departure curve. The delay function is given by

$$D(t) = \int_0^t d(x)\,dx \tag{7}$$

It is true that $D(t) \in \mathcal{F}$ as $D(t) = 0$ for $t <= 0$ and $D(t)$ is wide-sense increasing. Thus $D(t)$ features the same properties as arrival and service functions and can be described with similar algebraic methods.

In Figure 2 the arrival flow $R(t)$ grows faster than the service flow $C(t)$. Consequently, the departure flow $R^*(t)$ is delayed until $R(t) \leq C(t)$.

Equation 6 defines delay as time difference between equal values of arrival and departure function. The horizontal axis is the time axis, so the area D between $R(t)$ and $R^*(t)$ is the sum of occurring delays. Figure 3 shows delay function $D$ for a lower service curve $\beta^L$ instead of $C(t)$. A worst case scenario is given in Figure 4. It shows the overall delay $D(m)$ between $\alpha^U(t)$ and $(\alpha^U \otimes \beta^L)(t)$. By equation 6, $D(m)$ includes the area right from $m$ limited by $\alpha^U(m)$. Additionally Figure 4 includes delay function $D(t)$ as wide-sense increasing sum of occurring delays.

The last step is to define delay curves with a delay curve property. They will be the proposed formalization of quantitative requirements in SLAs.

*Definition 6.* An upper delay curve $\Psi^U$ for delay functions $D(t)$ satisfies the relation

$$\Psi^U(t-s) \geq D(t) - D(s)\ \forall\, 0 \leq s \leq t \tag{8}$$

$\Psi^U(t)$ is an upper bound of occurring delays in time interval $[0, t]$. Also a lower delay curve $\Psi^L$ is given by $\Psi^L \leq D(t) - D(s)\ \forall\, 0 \leq s \leq t$. Minimum delay times are infrequently used in computer science. In other modeling domains such as production systems and chemical processes some work processes might require to run a minimum amount of time. This work will focus on upper delay curves.

Concrete instances of upper delay curves can use same functions set as for arrival curves. For modeling quantitative requierements in SLAs usage of the T-SPEC function class is suggested. The T-SPEC parameters can be interpreted as follows:

- $p_\Psi$ is the peak rate of delay growth
- $r_\Psi$ is the sustainable delay rate and
- $b_\Psi$ is the tolerance for bursts of delays.

The delay function $D(t)$ in figure 4 complies to a upper delay curve $\Psi^U$ that forms a limit for every sum of delays. $\Psi^U$ is in T-SPEC function class. Also the numeric examples in section 5 are using the interpretation given above.

## 4. DERIVING SERVICE CURVES

When designing SOA systems a typical task for service providers is to dimension computer systems according to the arrangements specified in Service Level Agreements. An SLA could include $\alpha^U$ to bound the arrival rate of requests and an upper delay curve $\Psi^U$ to control response times. The service provider has to find a matching service curve $\beta^L$ to meet the SLA.

With $\alpha^U$ and $\Psi^U$ we are assuming the worst-case behavior of the system under the highest allowed load. On the one hand, each service curve $\beta^L$ that causes less than $\Psi^U(t)$ delays in $[0, t]$ belongs to a system that works faster than needed. In practice, the system would more processing power than necessary and would be oversized. One the other hand, every $\beta^L$ producing more than $\Psi^U$ delays in $[0, t]$ would result in a system that is too slow and does not comply with the specification.

In the optimal case we have $\Psi^U = D(t)$. When using T-SPEC functions for $\alpha^U$ and $\Psi^U$ identity is not achievable by matching piecewise linear segments with integral functions of higher grade. A relaxation of the condition to $\Psi^U(t) \geq D(t)$ allows us to set up an optimization problem to minimize target function $F$:

$$F = \Psi^U - D \text{ such that } \Psi^U(s) \geq D(s)\ \forall s \in [0, t]. \qquad (9)$$

In this work a method is presented to find to the closest approximation by reduction to an geometric optimization problem with just one variable.

Before going into details two mathematical concepts are introduced.

### 4.1 Pseudo-Inverse

Every strictly increasing function is left-invertible [3]:

$$\forall t_1 < t_2, f(t_1) < f(t_2)\ \exists f^{-1} \in \mathcal{F} : f^{-1}(f(t)) = t\ \forall t$$

In wide-sense increasing functions there might be the case of $f(a) = y = f(b)$ for $a < b$. With such plateaus functions

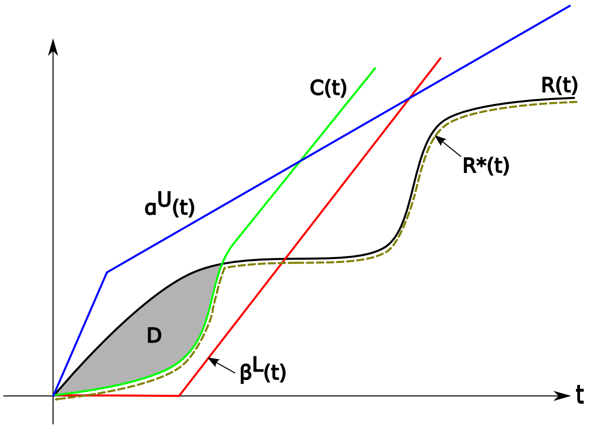

**Figure 2: Arrivals $R(t)$, service $C(t)$ and resulting delay $D(m)$**

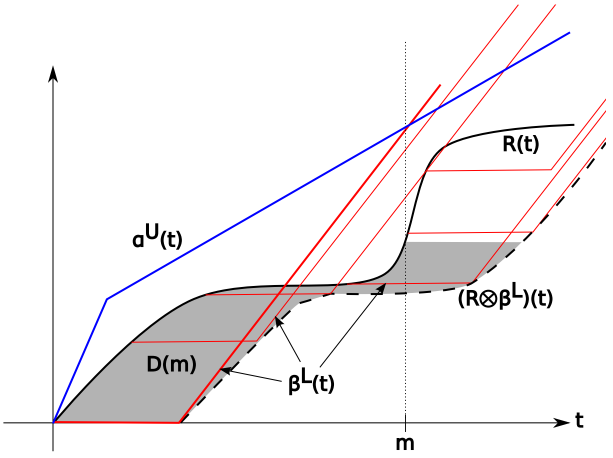

**Figure 3: Arrivals $R(t)$ and delay $D(m)$. Service curve $\beta^L(t)$ is moved along $R(t)$ for convolution, the result is the lower envelope off all outgoing arrival curves $(R \otimes \beta^L)(t)$ .**

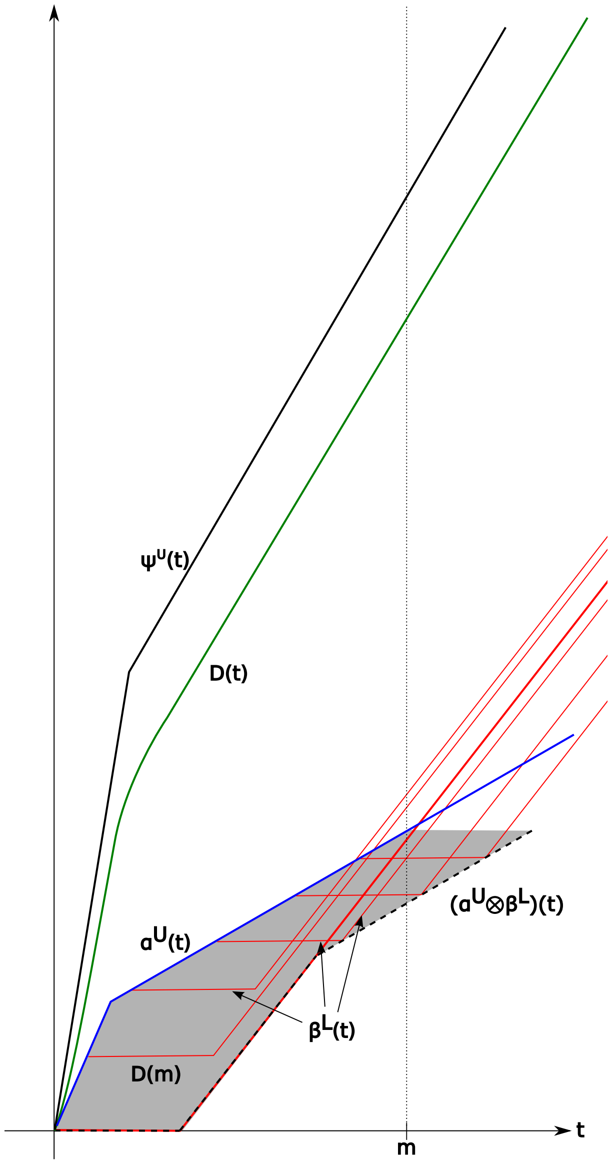

**Figure 4: Upper arrival curve $\alpha^U(t)$, resulting arrival curve $(\alpha^U \otimes \beta^L)(t)$ and delay $D(m)$. Graphs of delay function $D(t)$ and $\Psi^U(t)$ are not in scale to other functions.**

$f \in \mathcal{F}$ are not left-invertible. Otherwise there would be the situation where $f^{-1}(y) = a$ and $f^{-1}(y) = b$, $a \neq b$.
To solve this issue the concept of pseudo-inverse functions is used.

*Definition 7.* Pseudo-inverse $f^{-1}$ of $f \in \mathcal{F}$ [3]:

$$f^{-1}(x) = \inf\{t \text{ such that } f(t) \geq x\} \tag{10}$$

As T-SPEC and $\beta_{R,T}$ functions are in $\mathcal{F}$ the pseudo-inverse has to be used. For example, the inverse of $\beta_{R,T}$ is

$$\beta_{R,T}^{-1}(t) = \begin{cases} 0 & \text{for } t = 0 \\ \frac{t}{R} + T & \text{for } t > 0 \end{cases}$$

## 4.2 Horizontal Deviation

The vertical deviation between two functions $f, g$: $v_{f,g}(t) = |f(t) - g(t)|$ is the difference at the same input. It is used in [3] to calculate the bounds of backlogging for a system:

$$v(f,g) = \sup_{t \geq 0}\{f(t) - g(t)\} \tag{11}$$

It should also be noted that integration for areas $A = \int(f - g)$ between two functions uses vertical deviation.
Horizontal deviation between two functions is the difference in input to get the same output.

$$h_{f,g} = \inf\{d \geq 0 \text{ such that } f(t) \leq g(t+d)\} \tag{12}$$

Delay function $d(t)$ in section 3 is based on horizontal deviation. To form a delay function $D(t) = \int_0^t d(x)\, dx$ one would have to integrate with horizontal deviation. This can be done by converting the horizontal case to a vertical one. The transformation makes use of the pseudo-inverse.

Theorem 1 (horizontal deviation). *Let $f, g \in \mathcal{F}$.*

$$h_{f,g}(t) = g^{-1}(f(t)) - t \tag{13}$$

Proof. We start from (12) defining the horizontal deviation and apply Definition 7 of the pseudo-inverse.

$$\begin{aligned} h_{f,g}(t) &= \inf\{d \geq 0 : f(t) \leq g(t+d)\} \\ &= \inf\{d + t \geq 0 : g(t+d) \geq f(t)\} - t \\ &= \inf\{\Delta \geq 0 : g(\Delta) \geq f(t)\} - t && d + t = \Delta \\ &= g^{-1}(f(t)) - t && \text{by Def. (7)} \end{aligned}$$

□

Transformation from vertical to horizontal deviation is now used to approximate the minimum service curve.

## 4.3 Approximation of service curve $\beta^L$

In this section a lower service curve $\beta^L$ is approximated for a system node that is required to serve $\alpha^U$ with a cumulated delay smaller than $\Psi^U$.
The case is considered where $\alpha^U(t)$ and the $D(t)$ are both given as T-SPEC arrival curves, lower service curve $\beta^L$ will be modeled as rate-latency function

$$\beta_{R,T}(t) = \max\{0, R[t - T]\}$$

Both function classes can model a wide set of arrival and service behaviors, especially in the domain of networks and SOA. The modeling approach can be extended in the future to similar piecewise linear functions for a closer approximation as done in [7] for realtime systems.

For optimization, target function (9) has to be parametrized with a variable that only influences the service function $\beta^L = \beta_{R,T}$ since $\Psi^U$ and $\alpha^U$ are fixed. First, the system latency $T$ will be derived from $\Psi^U$. Second, a relationship between an significant point of function $\alpha^U \otimes \beta^L$ and service rate $R$ is set. The position of this point is variated for optimization. Finally, a formula for $D(t)$ depending on the point is derived and used in the target function.

For system latency $T$ we can observe that delay curve $\Psi^U(t)$ includes the maximum allowed latency as sustainable rate.

Lemma 1 (maximum system latency). *When $\alpha^U$ = T-SPEC$(p_\alpha, 0, r_\alpha, b_\alpha)$, $\Psi^U(t)$ = T-SPEC$(p_\Psi, 0, r_\Psi, b_\Psi)$, $\beta^L = \beta_{R,T}$ with $R > r_\alpha$ then $r_\Psi$ is the upper bondary of system latency $T$.*

Proof. Consider a system with a delay $T > 0$ and infinite service rate. It has a service curve described by the burst-delay function $\delta_T(t) = \beta_{\infty,T} = \infty$ for $t > T$ and 0 otherwise. We can derive the departure curve by $\alpha^{L'}(t) = (\alpha^U \otimes \delta_T)(t) = \alpha^U(t-T)$. This equals shifting $\alpha^U$ by $T$ to the right. Hence, the horizontal deviation is $h_{\alpha^U,\alpha^{L'}}(t) = T \ \forall t$. For the cumulated delay we have $D(t) = \int_0^t T\,dx = Tt$, the slope of delay function is $T$. It complies with delay curves with a sustainable rate $r_\Psi \geq T$. $\square$

As we are searching for the lowest service curve we will fix the worst-case system latency $r_\Psi = T$ for service curve $\beta_{R,T}$.

Now the service rate R in $\beta_{R,T}$ is formulated as a function depending on a discontinuous point of departure curve $\alpha^{L'} = \alpha^U \otimes \beta_{R,T}$. We form the departure curve by (min,+)-convolution, $\alpha^U$ = T-SPEC$(\alpha_p, 0, \alpha_r, \alpha_b)$ and $\beta^L = \beta_{R,T}$. For details in computation we refer to [3], p. 111.

$$(\alpha^U \otimes \beta_{R,T})(t) = \begin{cases} 0 & 0 \leq t \leq T \\ R(t-T) & T < t < p \\ \alpha_r(t-T) + \alpha_b & t \geq p \end{cases} \tag{14}$$

Value $p$ is the intersection of both affine segments. We can make the following observations:

1. The departure curve ascends at point $p_1 = [T, 0]$ and continues to $p_2 = [p, \alpha^{L'}(p)]$
2. For $t \geq p$ the departure curve has a slope identical to the arrival curve.
3. The distance to $\alpha^U$ at $p$ is $h_{\alpha^U,\alpha^{L'}}(p-T) = T$.

Hence all possible $p_2$ are located on a curve that can be derived from $\alpha^U$ by shifting it $T$ to the right: $p_2 = [t, \alpha^U(t-T)]$. As a a result, the slope between $p_1$ and $p_2$ is

$$R = m(p) = \frac{\alpha^U(p-T)}{p-T} \text{ for } T < p \tag{15}$$

Now $\beta^L(t)$ and $(\alpha^U \otimes \beta^L)(t)$ receive additional parameter $p$: $\beta_{R,T}(p,t) = \max\{0, m(p)(t-T)\}$ and for the convolution

$$(\alpha^U \otimes \beta^L(p))(p,t) = \begin{cases} 0 & 0 \leq t \leq T \\ m(p)(t-T) & T < t < p \\ \alpha_R(t-T) + \alpha_B & t \geq p \end{cases} \tag{16}$$

For the sum of delays we use integration for horizontal deviation and also parametrize it with $p$:

$$\begin{aligned} D(p,t) &= \int_0^t h(\alpha^U, \alpha^U \otimes \beta^L(p))(x)\,dx \\ &= \int_0^t (\alpha^U \otimes \beta^L(p))^{-1}(\alpha^U(x)) - x\,dx \quad \text{using (13)} \end{aligned}$$

Pseudo-inverse of the parametrized departure function:

$$(\alpha^U \otimes \beta^L)^{-1}(p,t) = \begin{cases} \frac{t}{m(p)} + T & 0 \leq t \leq \alpha^U(p-T) \\ \frac{t}{\alpha_R} - \frac{\alpha_B}{\alpha_R} + T & t > \alpha^U(p-T) \end{cases}$$

The antiderivative for the horizontal deviation uses the substitution rule for linear functions, $\alpha^{U'}$ is derivative:

$$D(p,t) = [(\alpha^U \otimes \beta^L(p))^{-1}(\alpha^U(t)) - t]_0^t \tag{17}$$

$$= \frac{s(p,t)}{\alpha^{U'}(t)} - \frac{t^2}{2} \tag{18}$$

$$s(p,t) = \begin{cases} \frac{t^2}{2m(p)} + Tx & 0 \leq t \leq p-T \\ \frac{x^2}{2\alpha_R} - \frac{\alpha_B x}{\alpha_R} + Tx & t > p-T \end{cases} \tag{19}$$

Function $\alpha^{U'}(x)$ has also a discontinuity at $x = \frac{\alpha_B}{\alpha_P - \alpha_R}$ that has to be considered when computing the integral.

$$\alpha^{U'}(t) = \begin{cases} \alpha_p & 0 \leq t \leq \frac{\alpha_B}{\alpha_P - \alpha_R} \\ \alpha_r & t > \frac{\alpha_B}{\alpha_P - \alpha_R} \end{cases} \tag{20}$$

Finally the target function is rewritten to

$$F(p) = \Psi^U - D(p) \text{ such that } \Psi^U(s) \geq D(p,s) \ \forall s \in [0,t].$$

## 5. EXAMPLE

The optimization problem to find a lower service curve from quantitative requirements in SLAs presented above has been implemented in Matlab. To minimize cost function $F(p)$ the build-in Matlab command `fminsearch` is used.

For illustration consider the planning of a simple server system answering customer requests. Arriving requests are enqueued in a waiting queue if the server is busy and the turnaround time is measured as delay. Normally the server will process all request at maximum system speed, but sometimes it runs some maintenance tasks that degrade its service level for a short time and enforce backlogging of requests.

An SLA using a T-SPEC arrival curve limits the arrival rate and thus the load to the system to $\alpha_r$, but for a short time a peak rate of $\alpha_p$ is also accepted. The SLA enforces a quantitative requirement on delay times. A delay curve, again of T-SPEC type, limits sustainable growth rate of aggregated waiting times per time period to $r_\Psi = 2.0$. To accept possible extra delays caused by server maintenance the SLA grants a permission to let the delays sum grow at rate $r_\Psi = 3.5$ for a limited phase. With this information a minimum service curve $\beta^L = \beta_{R,T}$ is derived by solving the optimization task described above.

**Table 1: Results for $\Psi$ = T-SPEC$(3.5, 0, 2.0, 15)$**

| $\alpha_b = 3$ | | $\alpha_r$ | | | |
|---|---|---|---|---|---|
| | | 0.5 | 0.75 | 1.0 | 1.25 |
| $\alpha_p$ | 1.5 | 0.8224 | 0.9658 | 1.1575 | 1.3664 |
| | 2.0 | 0.9019 | 1.0204 | 1.1982 | 1.4135 |
| | 2.5 | 0.8117 | 1.1021 | 1.2107 | 1.4281 |
| | 3.0 | 0.8184 | 1.1437 | 1.2413 | 1.4382 |

**Table 2: Results for $\Psi$ = T-SPEC$(5.0, 0, 1.5, 20)$**

| $\alpha_b = 3$ | | $\alpha_r$ | | | |
|---|---|---|---|---|---|
| | | 0.5 | 0.75 | 1.0 | 1.25 |
| $\alpha_p$ | 1.5 | 0.7315 | 0.9304 | 1.1401 | 1.3494 |
| | 2.0 | 0.7371 | 0.9465 | 1.1602 | 1.3822 |
| | 2.5 | 0.7544 | 0.9569 | 1.1681 | 1.3912 |
| | 3.0 | 0.7594 | 0.9609 | 1.1723 | 1.3972 |

Table 1 shows for the example the minimum rate $R$ in $\beta_{R,T}$ for 16 different arrival curves $\alpha^U$ = T-SPEC$(\alpha_p, 0, \alpha_r, \alpha_b)$ under the quite strict delay curve taken from the SLA. Service rate $R$ has to grow with the arrival rate to stay below the restrictions of $\Psi^U$.
For comparison, table 2 contains the minimum sustainable rates for $\beta_{R,T}$ using the same arrival conditions but a different delay curve $\Psi$ = T-SPEC$(5.0, 0, 1.5, 20)$ which is a more relaxed SLA requirement. As a result, service rate $R$ can be chosen much lower and allows the selection of a slower and cheaper service provider.

## 6. CONCLUSIONS AND FUTURE WORK

In this paper a new method to model quantitative requirements in SLAs was introduced. It makes use of and extends (min,+)-based queueing system theory.
(min,+) algebra used in Network Calculus and Real-Time Calculus was introduced briefly. Both approach model system load with arrival functions and set limits in form of arrivals curves. In a similar way service curves describe available processing power. With (min,+) convolution the system departure curve can be bounded.
For descriptions of quantitative requirements in SLAs with (min,+) systems the new concept of delay curves was introduced. Delays as horizontal deviation between arrival and departure curves are integrated to delay functions. As these functions are comparable to arrival functions they can also be bounded with delay curves. The novel delay curves are used to describe SLAs with tolerances more accurate than definitions of a hard deadlines.
As use case for delay curves the minimum service rate a service provider has to offer is computed from given delay and arrival curves. The necessary service curve is approximated by solving an optimization problem.

Future research will deploy delay curves for validation of quantitative requirements of SLAs in SOA systems as described in [2]. For SOA the system node model in figure 1 with delay curves will be extend for networks of nodes. Similar to remaining service curves [7] used in Real-Time Calculus delay curves of nodes are influenced when processing is done in serial or parallel. Further algebraic relationships and theorems for delay curves will be refined. For software solutions [2] best practice advises for translation of SLAs into a set of curves will be investigated.

## 7. REFERENCES

[1] F. Baccelli, G. Cohen, G. Olsder, and J. Quadrat. *Synchronization and linearity.* Wiley New York, 1992.

[2] F. Bause, P. Buchholz, J. Kriege, and S. Vastag. Simulation based validation of quantitative requirements in service oriented architectures. In M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls, editors, *Proceedings of the 2009 Winter Simulation Conference*, pages 1015–1026. IEEE, 2009.

[3] J.-Y. L. Boudec and P. Thiran. *Network Calculus - A Theory of Deterministic Queuing Systems for the Internet*, volume 4 of *LNCS*. Springer Verlag, May 2004.

[4] C. Chang. Performance guarantees in communication networks. *European Transactions on Telecommunications*, 12(4):357–358, 2001.

[5] H. Sarjoughian, S. Kim, M. Ramaswamy, and S. Yau. A simulation framework for service-oriented computing systems. In S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, and J. W. Fowler, editors, *Proceedings of the 2008 Winter Simulation Conference*, pages 845–853, Piscataway, New Jersey, 2008. Institute of Electrical and Electronics Engineers, Inc.

[6] S. Shenker and J. Wroclawski. RFC 2215. *General Characterization Parameters for Integrated Service Network Elements*, 1997.

[7] L. Thiele, S. Chakraborty, M. Gries, and S. Künzli. Design space exploration of network processor architectures. *Network Processor Design: Issues and Practices*, 1:55–89, 2002.

[8] L. Thiele, S. Chakraborty, M. Gries, A. Maxiaguine, and J. Greutert. Embedded software in network processors—models and algorithms. In *Embedded Software*, pages 416–434. Springer, 2001.

[9] L. Thiele, S. Chakraborty, and M. Naedele. Real-time calculus for scheduling hard real-time systems. In *Circuits and Systems, 2000. Proceedings. ISCAS 2000 Geneva. The 2000 IEEE International Symposium on*, volume 4, 2000.

[10] J. J. M. Trienekens, J. J. Bouman, and M. van der Zwan. Specification of service level agreements: Problems, principles and practices. *Software Quality Journal*, 12(1):43–57, 2004.

[11] W. M. P. van der Aalst. Using interval timed coloured petri nets to calculate performance bounds. In *Proceedings of the 7th international conference on Computer performance evaluation : modelling techniques and tools*, pages 425–444, Secaucus, NJ, USA, 1994. Springer-Verlag New York, Inc.

[12] S. Vastag. Modellvalidierung von zeitbegrenzten logistischen prozessketten mit interval timed coloured petri nets. In S. Fischer, E. Maehle, and R. Reischuk, editors, *Proceedings of INFORMATIK 2009*, volume 154 of *Lecture Notes in Informatics*, page 465, Bonn, September 2009. Gesellschaft für Informatik.